**Tugas 2 (PL/SQL)**

**PL/SQL**

*Procedural Language/Structured Query Language* (PL/SQL) merupakan sebuah penggabungan antara bahasa pemrograman prosedural (PL) dan SQL *syntax*. Berbeda dengan SQL yg hanya mementingkan bagaimana mendapatkan hasil, dengan PL/SQL juga akan diketahui bagaimana proses untuk mendapatkan hasil tersebut.

Dalam PL/SQL dapat digunakan perintah untuk memanipulasi data yang ada dalam database MySQL ataupun Oracle dan membentuk pemrograman terstruktur dalam memproses data. Pada PL/SQL ditambahkan beberapa hal yang dikenal pada dunia pemrograman, seperti variabel, loop, pemrosesan kondisi, operasi cursor, modularisasai, dan hal-hal lainnya. Semua tambahan itu bertujuan untuk meningkatkan kinerja operasi-operasi SQL pada *database* sehingga manfaat dari kehandalannya menjadi maksimal.

PL/SQL selalu terdiri dari 3 (tiga) bagian utama, yaitu deklarasi (*DECLARE*), pemrograman (*BEGIN…END*), serta pengendalian *Error* (*EXCEPTION*).

Bagian utama yang wajib ada dalam satu blok PL/SQL adalah pemrograman, sedangkan bagian deklarasi dan pengendalian *Error* bersifat opsional.Structure penulisan pada PL/SQL :

DECLARATE

BEGIN

END;

Ex: PL/SQL------------------

DECLARATE

x NUMBER;

BEGIN

x := 5;

DBMS\_OUTPUT.PUT\_LINE(x);

END;

// output : 5

**STORED PROCEDURE**

PL/SQL secara umum terbagi atas 2 jenis, yaitu Anonymous Block dan Stored Block (Procedure).

Anonymous Block adalah kode PL/SQL yang tidak tersimpan ke dalam Database.  
Blok kode ini bisa melakukan pemrograman selayaknya PL/SQL, namun kode ini tidak bisa dipanggil/diakses kembali (kecuali code di simpan ke laptop / komputer).

Stored Block, atau lebih dikenal dengan procedure, adalah blok kode PL/SQL yang tersimpan di dalam Database, dan bisa dipanggil kembali dengan menggunakan nama dari procedure tersebut.

Procedure melakukan tugas tanpa mereturn value, seperti method void pada pemrograman Objek Oriented Programming C#.

CREATE OR REPLACE PROCEDURE hello IS

BEGIN

DBMS\_OUTPUT.PUT\_LINE(‘Hi’);

END;

// run -------------------------

BEGIN

hello();

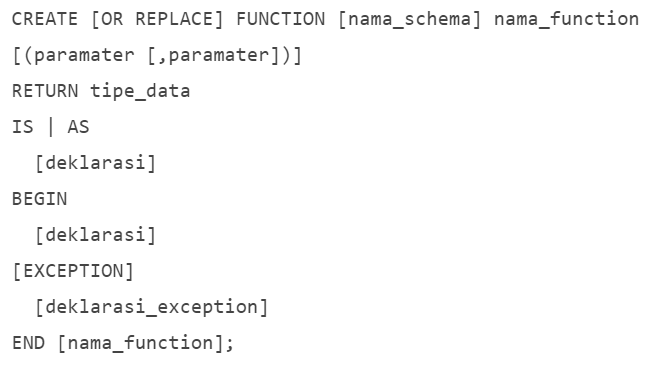
END;

// output : Hi

**FUNCTION**

Function adalah suatu blok PL/SQL yang memiliki konsep sama dengan procedure, hanya saja pada function terdapat pengembalian nilai (return value). Karena function dapat mengembalikan sebuah nilai, function dapat diakses seperti layaknya sebuah variabel biasa.

Untuk membuat Function di PL/SQL, dapat menggunakan perintah CREATE OR REPLACE FUNCTION. berikut dibawah adalah sintax dasar untuk membuat function sql.

****

**contoh membuat function SQL :**

**Contoh 1 :**

**CREATE OR REPLACE FUNCTION LINGKARAN (jari number)**

**RETURN NUMBER IS**

**BEGIN**

**return((22/7\*jari\*jari)+(2\*22/7\*jari));**

**END LINGKARAN;**

**/**

**Declare**

**temp number;**

**jari2 number;**

**Begin**

**jari2 := 7;**

**temp := LINGKARAN(jari2);**

**DBMS\_OUTPUT.PUT\_LINE('Lingkaran dengan jari jari '|| jari2**

**||' memliki jumlah luas dan keliling sebesar '|| temp);**

**End;**

**/**

**Contoh 2 :**

**CREATE OR REPLACE FUNCTION difference (x in NUMBER, y IN NUMBER, z IN NUMBER)**

**RETURN NUMBER IS**

**BEGIN**

**Z = X - Y;**

**RETURN Z;**

**END;**

**DECLARE**

**x NUMBER = 10;**

**y NUMBER = 5;**

**z NUMBER;**

**BEGIN**

**DBMS\_OUTPUT.PUT\_LINE(difference(x,y,z))**

**END;**

**TRIGGER**

Trigger adalah blok PL/SQL atau prosedur yang berhubungan dengan table, view, skema atau database yang dijalankan secara implicit pada saat terjadi sebuah event. Trigger merupakan store procedure yang dijalankan secara automatis saat user melakukan modifikasi data pada tabel. Modifikasi data yang dilakukan pada tabel yaitu berupa perintah INSERT, UPDATE, dan DELETE. INSERT , UPDATE dan DELETE bisa digabung jadi satu trigger yang dinamakan Multiple Trigger.

Beberapa penggunaan trigger misalnya:

– Membuat log untuk mencatat penggunaan tabel

– Mengupdate tabel lain jika terjadi perubahan pada suatu tabel. misalnya terjadi pembelian maka ketika tabel pembelian terUPDATE, maka tabel stock akan berubah.

– Menjalankan perhitungan-perhitungan otomatis.

**Trigger perlu dibuat pada saat** :

· Membentuk sebuah aksi tertentu terhadap suatu event

· Memusatkan operasi global

**Contoh penulisan trigger**

CREATE OR REPLACE TRIGGER ( nama\_trigger)

AFTER INSERT ON (table tujuan)

FOR EACH ROW

DECLARE

X NUMBER;

BEGIN

X:=:NEW.(nama\_kolom);

UPDATE (table yang di ubah) SET (perhitungan)jml\_stok = jml\_stok - X ;

END;

**Contoh TRIGGER :**

**//buat table**

CREATE TABLE paket\_barang(

id\_paket VARCHAR(4) PRIMARY KEY

);

CREATE TABLE barang(

id\_paket VARCHAR(4),

id\_barang NUMBER PRIMARY KEY,

nama VARCHAR(255),

harga NUMBER,

FOREIGN KEY(id\_paket) REFERENCES paket\_barang(id\_paket)

);

//insert table yang akan di trigger

INSERT INTO paket\_barang(id\_paket) VALUES ('A001');

INSERT INTO barang VALUES ('A001',0,'Headset', 50000 );

INSERT INTO barang VALUES ('A001',1,'RAM', 250000 );

INSERT INTO barang VALUES ('A001',2,'Mouse', 100000 );

//blok trigger

CREATE OR REPLACE TRIGGER paket\_baru

AFTER INSERT ON paket\_barang

FOR EACH ROW

DECLARE

TYPE list\_angka IS VARRAY(3) OF NUMBER;

TYPE lis\_barang IS VARRAY(3) OF VARCHAR(255);

nomor\_barang list\_angka = list\_angka(0,1,2);

nama\_barang list\_barang = list\_barang('headset', 'RAM', 'Mouse');

harga\_barang list\_angka = list\_angka(50000, 250000,100000);

paket VARCHAR(4) = 'A001';

BEGIN

FOR x in 1..3 LOOP

INSERT INTO barang(id\_paket, id\_barang, nama, harga)

VALUES (paket, nomor\_barang(x), nama\_barang(x), harga\_barang(x));

END LOOP;

END;

**SCHEDULER/EVENTS**

The Scheduler example here creates a simple job that runs a stored Programming Language/Structured Query Language (PL/SQL) procedure. The procedure selects a count of the number of users on the system and inserts that number into a table with a timestamp. It runs every five minutes.

https://gist.github.com/hadipsy27/cf1342bc06bc56e06a1269ebc6d9a78d